Cars_Dataset_Analysis

Naveen Alakunta

2024-11-01

Stage 1: Data Loading and Initial Inspection

Loading the Dataset

Initially, we import the car dataset from a CSV file by using the read.csv function.

```
cars_dataset <- read.csv("C:/Users/Naveen/OneDrive/Desktop/R Files/cars_data_10K.csv")
```

Checking Dataset Structure and Summary

Using `str()` to examine data types and `summary()` for important statistics.

```
str(cars_dataset)
```

```
## 'data.frame':    10000 obs. of  16 variables:
##  $ Make             : chr  "Chevrolet" "Lexus" "Suzuki" "Land Rover" ...
##  $ Model            : chr  "Black Diamond Avalanche" "RX 330" "Sidekick" "Range Rover" ...
##  $ Year             : int  2013 2005 1996 2016 2015 2015 2010 2017 2015 2006 ...
##  $ Engine.Fuel.Type : chr  "flex-fuel (unleaded/E85)" "regular unleaded" "regular unleaded" "diesel"
##  $ Engine.HP        : int  320 230 120 254 321 175 161 184 281 150 ...
##  $ Engine.Cylinders : int  8 6 4 6 6 4 4 4 6 4 ...
##  $ Transmission.Type: chr  "AUTOMATIC" "AUTOMATIC" "MANUAL" "AUTOMATIC" ...
##  $ Driven_Wheels    : chr  "four wheel drive" "all wheel drive" "four wheel drive" "four wheel drive
##  $ Number.of.Doors  : int  4 4 4 4 2 4 4 4 4 4 ...
##  $ Market.Category  : chr  "Crossover,Flex Fuel" "Crossover,Luxury" "N/A" "Diesel,Luxury" ...
##  $ Vehicle.Size     : chr  "Large" "Midsize" "Compact" "Large" ...
##  $ Vehicle.Style    : chr  "Crew Cab Pickup" "4dr SUV" "4dr SUV" "4dr SUV" ...
##  $ highway.MPG      : int  21 22 23 29 28 34 28 36 23 24 ...
##  $ city.mpg         : int  15 16 20 22 18 22 20 23 16 17 ...
##  $ Popularity       : int  1385 454 481 258 1624 5657 586 1013 1385 1851 ...
##  $ MSRP             : int  47885 37425 2000 93450 48165 22500 21700 25690 35795 18630 ...
```

```
summary(cars_dataset)
```

```
##      Make              Model                Year      Engine.Fuel.Type
##  Length:10000       Length:10000       Min.   :1990   Length:10000
##  Class :character   Class :character   1st Qu.:2007   Class :character
##  Mode  :character   Mode  :character   Median :2015   Mode  :character
```

```
##                                        Mean   :2010
##                                        3rd Qu.:2016
##                                        Max.   :2017
##
##    Engine.HP    Engine.Cylinders Transmission.Type  Driven_Wheels
##  Min.   :  55   Min.   : 0.000   Length:10000       Length:10000
##  1st Qu.: 170   1st Qu.: 4.000   Class :character   Class :character
##  Median : 227   Median : 6.000   Mode  :character   Mode  :character
##  Mean   : 249   Mean   : 5.632
##  3rd Qu.: 300   3rd Qu.: 6.000
##  Max.   :1001   Max.   :16.000
##  NA's   :62     NA's   :25
##  Number.of.Doors Market.Category    Vehicle.Size       Vehicle.Style
##  Min.   :2.000   Length:10000       Length:10000       Length:10000
##  1st Qu.:2.000   Class :character   Class :character   Class :character
##  Median :4.000   Mode  :character   Mode  :character   Mode  :character
##  Mean   :3.434
##  3rd Qu.:4.000
##  Max.   :4.000
##  NA's   :3
##   highway.MPG       city.mpg       Popularity        MSRP
##  Min.   : 12.00   Min.   :  7.0   Min.   :   2   Min.   :   2000
##  1st Qu.: 22.00   1st Qu.: 15.0   1st Qu.: 549   1st Qu.:  20960
##  Median : 25.00   Median : 18.0   Median :1385   Median :  29935
##  Mean   : 26.59   Mean   : 19.7   Mean   :1558   Mean   :  40341
##  3rd Qu.: 30.00   3rd Qu.: 22.0   3rd Qu.:2009   3rd Qu.:  42146
##  Max.   :354.00   Max.   :137.0   Max.   :5657   Max.   :1705769
##
```

Standardizing Column Names

Substitute dashes with dots in column titles for uniformity.

```r
names(cars_dataset) <- gsub("_", ".", names(cars_dataset))
```

Stage 2: Data Cleaning

1.Figuring out Lacking Values and Blank Strings:

We begin by checking the dataset for any missing values and incorrect entries, such as:

Checking for null (NA) values

This detects columns containing typical missing values (NA).

```r
colSums(is.na(cars_dataset))
```

```
##              Make             Model              Year   Engine.Fuel.Type
##                 0                 0                 0                  0
##         Engine.HP  Engine.Cylinders Transmission.Type      Driven.Wheels
```

```
##               62              25               0               0
##  Number.of.Doors Market.Category    Vehicle.Size   Vehicle.Style
##                3               0               0               0
##      highway.MPG        city.mpg      Popularity            MSRP
##                0               0               0               0
```

Verifying if strings are empty

Find blank values in the 'Engine.Fuel.Type' column, indicating potential missing data.

```r
sum(cars_dataset$Engine.Fuel.Type == "")
```

```
## [1] 3
```

```r
# Empty Strings found only in Engine.Fuel.Type
```

Identifying for N/A strings

Search for instances of "N/A" in Market.Category, as they might require conversion to typical NA values.

```r
sum(cars_dataset$Market.Category == "N/A")
```

```
## [1] 3196
```

```r
# N/A Strings found only in Market.Category
```

2..Dealing with missing information (NA's):

Once we have found any missing values, we go ahead and use different imputation methods to fill in these gaps.

Engine.HP

Incomplete data points are replaced with the median horsepower, providing a fair approximation that prevents any bias in the data.

```r
cars_dataset$Engine.HP[is.na(cars_dataset$Engine.HP)] <- median(cars_dataset$Engine.HP, na.rm = TRUE)
```

Engine.Cylinders

Similarly, any missing data points are substituted with the median number of cylinders.

```r
cars_dataset$Engine.Cylinders[is.na(cars_dataset$
      Engine.Cylinders)] <- median(cars_dataset$Engine.Cylinders, na.rm = TRUE)
```

Number.of.Doors

Unavailable data are substituted with the mode (4 doors), representing the most frequently seen arrangement.

```
cars_dataset$Number.of.Doors[is.na(cars_dataset$Number.of.Doors)] <- 4
```

Market.Category

We make missing entries uniform by changing "N/A" to NA and then substituting them with "Unknown" to ensure consistent categorization.

```
cars_dataset$Market.Category[cars_dataset$Market.Category == "N/A"] <- NA
cars_dataset$Market.Category[is.na(cars_dataset$Market.Category)] <- "Unknown"
```

3.Handling Empty Strings:

Engine.Fuel.Type

We substitute any blank entries in the Engine.Fuel.Type column with "Unknown" to maintain a uniform categorization of values.

```
cars_dataset$Engine.Fuel.Type[cars_dataset$Engine.Fuel.Type == ""] <- "Unknown"
```

4.Dealing with Outliers:

The Interquartile Range (IQR) method is used to address anomalies in important numerical columns in order to maintain the integrity of the analysis.

MSRP (Price)

```
Q1_MSRP <- quantile(cars_dataset$MSRP, 0.25)
Q3_MSRP <- quantile(cars_dataset$MSRP, 0.75)
IQR_MSRP <- Q3_MSRP - Q1_MSRP
cars_dataset <- subset(cars_dataset, MSRP >= (Q1_MSRP - 1.5 * IQR_MSRP) &
                         MSRP <= (Q3_MSRP + 1.5 * IQR_MSRP))
```

Engine.HP

```
Q1_HP <- quantile(cars_dataset$Engine.HP, 0.25)
Q3_HP <- quantile(cars_dataset$Engine.HP, 0.75)
IQR_HP <- Q3_HP - Q1_HP
cars_dataset <- subset(cars_dataset, Engine.HP >= (Q1_HP - 1.5 * IQR_HP) &
                         Engine.HP <= (Q3_HP + 1.5 * IQR_HP))
```

highway.MPG

```r
Q1_hwy <- quantile(cars_dataset$highway.MPG, 0.25)
Q3_hwy <- quantile(cars_dataset$highway.MPG, 0.75)
IQR_hwy <- Q3_hwy - Q1_hwy
cars_dataset <- subset(cars_dataset, highway.MPG >= (Q1_hwy - 1.5 * IQR_hwy) &
                         highway.MPG <= (Q3_hwy + 1.5 * IQR_hwy))
```

city.mpg

```r
Q1_city <- quantile(cars_dataset$city.mpg, 0.25)
Q3_city <- quantile(cars_dataset$city.mpg, 0.75)
IQR_city <- Q3_city - Q1_city
cars_dataset <- subset(cars_dataset, city.mpg >= (Q1_city - 1.5 * IQR_city) &
                         city.mpg <= (Q3_city + 1.5 * IQR_city))
```

Engine.Cylinders

```r
Q1_cylinders <- quantile(cars_dataset$Engine.Cylinders, 0.25, na.rm = TRUE)
Q3_cylinders <- quantile(cars_dataset$Engine.Cylinders, 0.75, na.rm = TRUE)
IQR_cylinders <- Q3_cylinders - Q1_cylinders
cars_dataset <- subset(cars_dataset, Engine.Cylinders >= (Q1_cylinders - 1.5 * IQR_cylinders) &
                         Engine.Cylinders <= (Q3_cylinders + 1.5 * IQR_cylinders))
```

5.Correcting Unrealistic Values:

Get rid of any rows in which Engine.HP or Engine.Cylinders have a value of zero or less

Rows with values of zero or less are deleted because they are not feasible for these characteristics.

```r
cars_dataset <- subset(cars_dataset, Engine.HP > 0)

cars_dataset <- subset(cars_dataset, Engine.Cylinders > 0)
```

Remove the rows with city.mpg and highway.MPG values less than 1 or greater than 100.

Limiting fuel efficiency values to a realistic range of 1 to 100 mpg is done to maintain accuracy of data and eliminate impractical entries.

```r
cars_dataset <- subset(cars_dataset, city.mpg > 0 & city.mpg <= 100)

cars_dataset <- subset(cars_dataset, highway.MPG > 0 & highway.MPG <= 100)
```

Setting rows with a Number of Doors ranging from 2 to 5.

Limiting values to fall within the standard range of 2 to 5 doors.

```
cars_dataset <- subset(cars_dataset, Number.of.Doors >= 2 & Number.of.Doors <= 5)
```

6.Transforming/Converting Data Types:

For data consistency, we change certain columns to suitable data types

Year

Convert from categorical to integer type to enable numerical analysis.

```
cars_dataset$Year <- as.integer(as.character(cars_dataset$Year))
```

Conversion to factors

List of categorical columns designated for conversion to factors.

```
categorical_columns <- c("Make", "Model", "Engine.Fuel.Type", "Transmission.Type",
                         "Driven.Wheels", "Market.Category", "Vehicle.Size", "Vehicle.Style")
```

Transform categorical_columns into factors.

```
cars_dataset[categorical_columns] <- lapply(cars_dataset[categorical_columns], as.factor)
```

Commentary:

Data cleaning addresses missing values, outliers, and unrealistic values to improve data quality. This step fills missing entries using suitable imputation methods, removes extreme outliers from key variables, and standardizes data types. By handling these issues, we prepare a consistent and accurate dataset for reliable analysis in subsequent steps.

Stage 3: Comprehensive Exploratory Data Analysis (EDA)

Here, we use visualizations and summaries to explore key features and relationships in the dataset. Charts help examine variable distributions and interactions, uncovering patterns that inform our analysis.

Dataset Summary

We begin with a dataset summary to review basic statistics, typical values, ranges, and any anomalies, guiding our choice of visualizations for deeper insights.

```
summary(cars_dataset)
```

```
##          Make                      Model           Year
##  Chevrolet : 903   Silverado 1500    : 131   Min.   :1990
##  Ford      : 710   Tundra            : 122   1st Qu.:2005
##  Volkswagen: 635   F-150             : 103   Median :2014
```

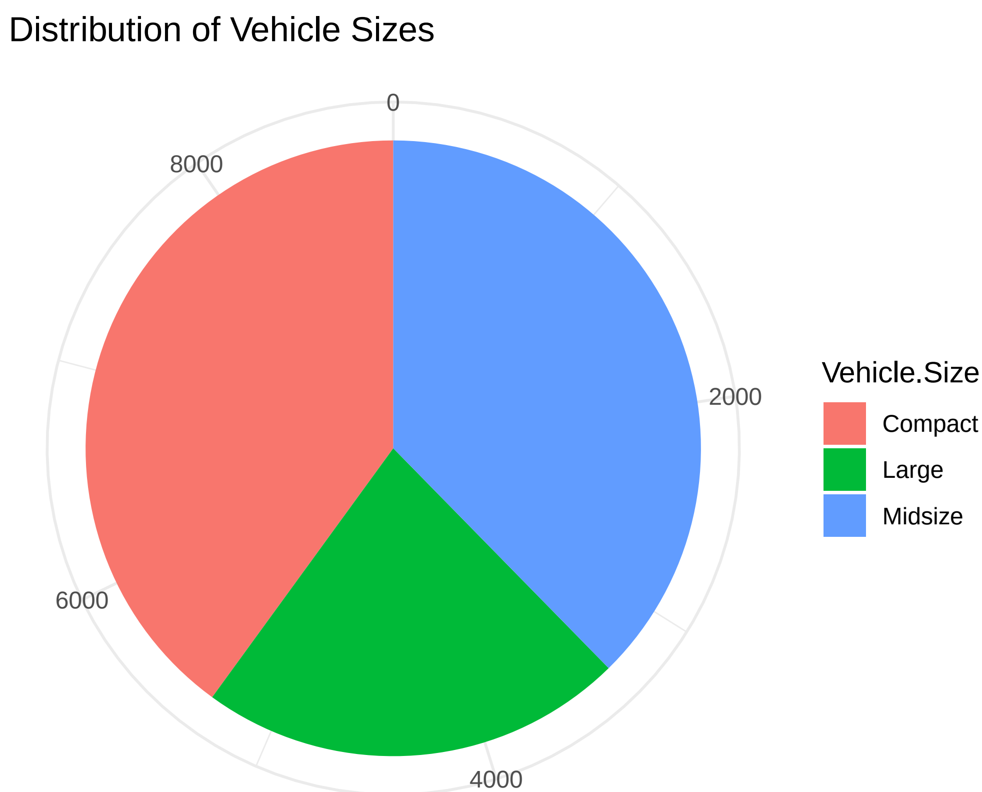
```
##  Toyota    : 579   Sierra 1500       :  74   Mean   :2010
##  Dodge     : 499   Beetle Convertible:  71   3rd Qu.:2016
##  Nissan    : 452   Frontier          :  65   Max.   :2017
##  (Other)   :5073   (Other)           :8285
##                                    Engine.Fuel.Type   Engine.HP
##  regular unleaded                          :5848    Min.   : 63.0
##  premium unleaded (recommended)            :1172    1st Qu.:170.0
##  premium unleaded (required)               : 962    Median :210.0
##  flex-fuel (unleaded/E85)                  : 722    Mean   :227.8
##  diesel                                    : 111    3rd Qu.:285.0
##  flex-fuel (premium unleaded recommended/E85):  22    Max.   :460.0
##  (Other)                                   :  14
##  Engine.Cylinders        Transmission.Type          Driven.Wheels
##  Min.   :3.0      AUTOMATED_MANUAL: 332    all wheel drive  :1672
##  1st Qu.:4.0      AUTOMATIC       :6244    four wheel drive :1115
##  Median :6.0      MANUAL          :2258    front wheel drive:3756
##  Mean   :5.4      UNKNOWN         :  17    rear wheel drive :2308
##  3rd Qu.:6.0
##  Max.   :8.0
##
##  Number.of.Doors          Market.Category  Vehicle.Size
##  Min.   :2.000   Unknown           :3184   Compact:3539
##  1st Qu.:3.000   Crossover         : 940   Large  :1979
##  Median :4.000   Flex Fuel         : 734   Midsize:3333
##  Mean   :3.482   Luxury            : 686
##  3rd Qu.:4.000   Performance       : 491
##  Max.   :4.000   Luxury,Performance: 486
##                  (Other)           :2330
##              Vehicle.Style   highway.MPG      city.mpg       Popularity
##  Sedan              :2210   Min.   :12.0   Min.   :10.00   Min.   :  21
##  4dr SUV            :1948   1st Qu.:22.0   1st Qu.:16.00   1st Qu.: 549
##  Coupe              : 700   Median :26.0   Median :18.00   Median :1385
##  Crew Cab Pickup    : 560   Mean   :26.3   Mean   :19.18   Mean   :1557
##  Extended Cab Pickup: 541   3rd Qu.:30.0   3rd Qu.:22.00   3rd Qu.:2009
##  4dr Hatchback      : 487   Max.   :43.0   Max.   :31.00   Max.   :5657
##  (Other)            :2405
##       MSRP
##  Min.   : 2000
##  1st Qu.:19795
##  Median :28395
##  Mean   :28433
##  3rd Qu.:38100
##  Max.   :73905
##
```

```r
#install.packages("ggplot2")
library(ggplot2)
```

1. Pie Chart for Vehicle Size Distribution

Shows the proportion of each vehicle size (e.g., Compact, Midsize), helping to understand the diversity and commonality of car sizes in the dataset.

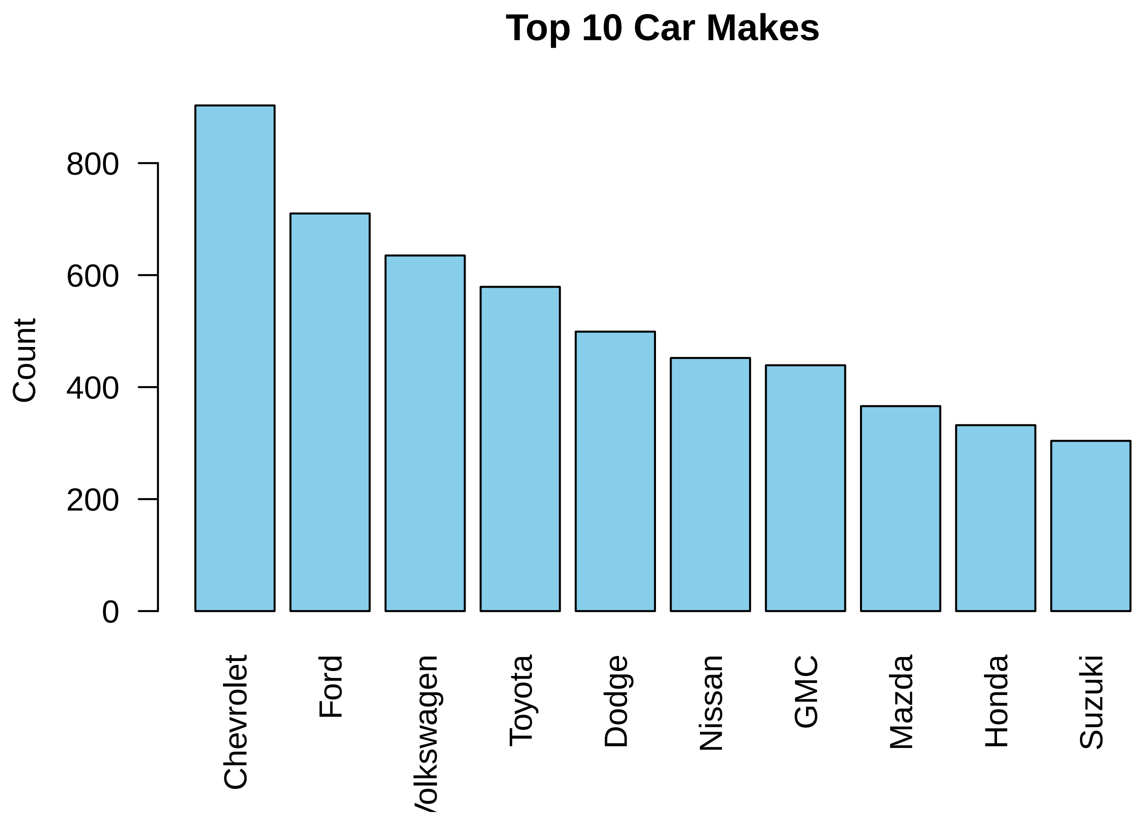
```r
ggplot(cars_dataset, aes(x = "", fill = Vehicle.Size)) +
  geom_bar(width = 1) +
  coord_polar("y") +
  labs(title = "Distribution of Vehicle Sizes") +
  theme_minimal() +
  theme(axis.title.x = element_blank(), axis.title.y = element_blank())
```



2. Bar Chart for Most Common Car Makes

Spotlights the top 10 most common car brands, providing insights into brand popularity and market share.

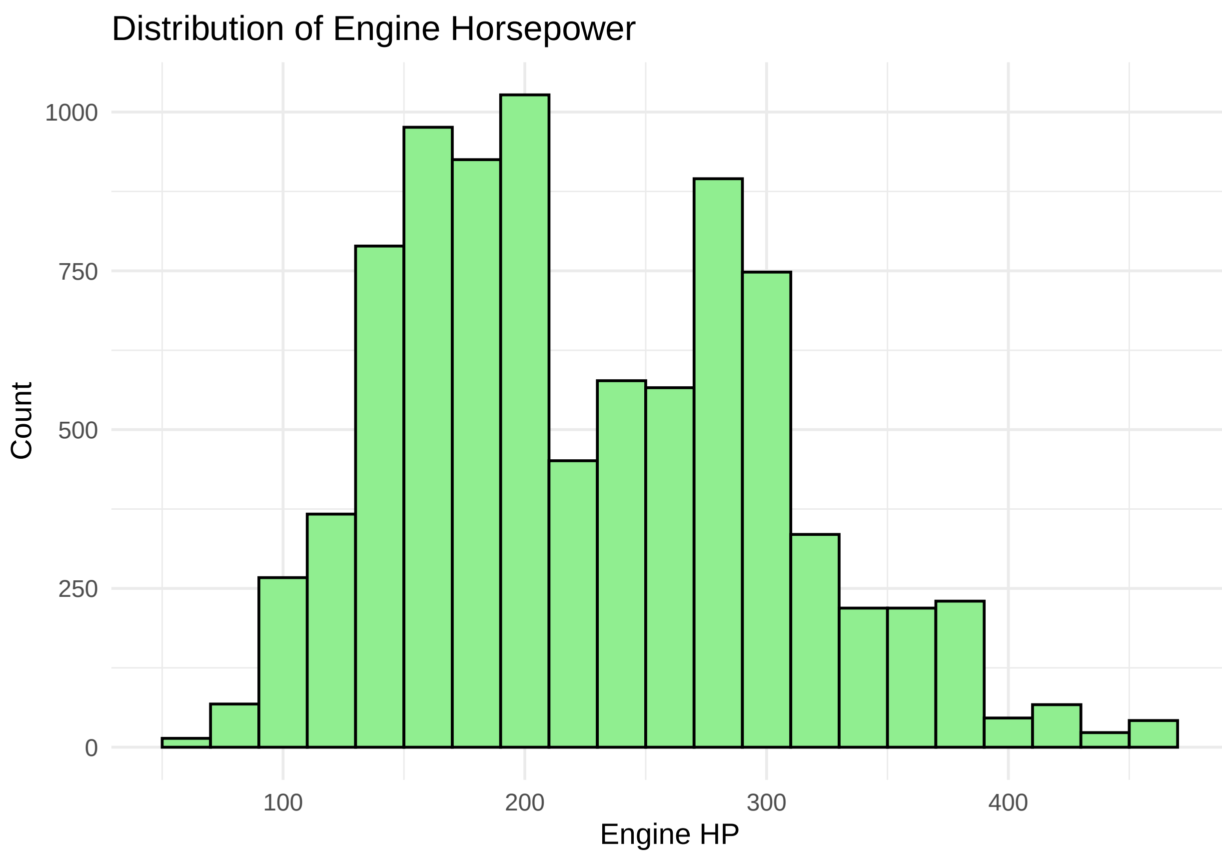
```r
top_10_makes <- sort(table(cars_dataset$Make), decreasing = TRUE)[1:10]
barplot(top_10_makes,
        main = "Top 10 Car Makes",
        col = "skyblue",
        las = 2,
        ylab = "Count")
```



3. Histogram of Engine Horsepower (Engine.HP)

Displays the distribution of horsepower, showing common performance levels and any trends in engine power across the dataset.

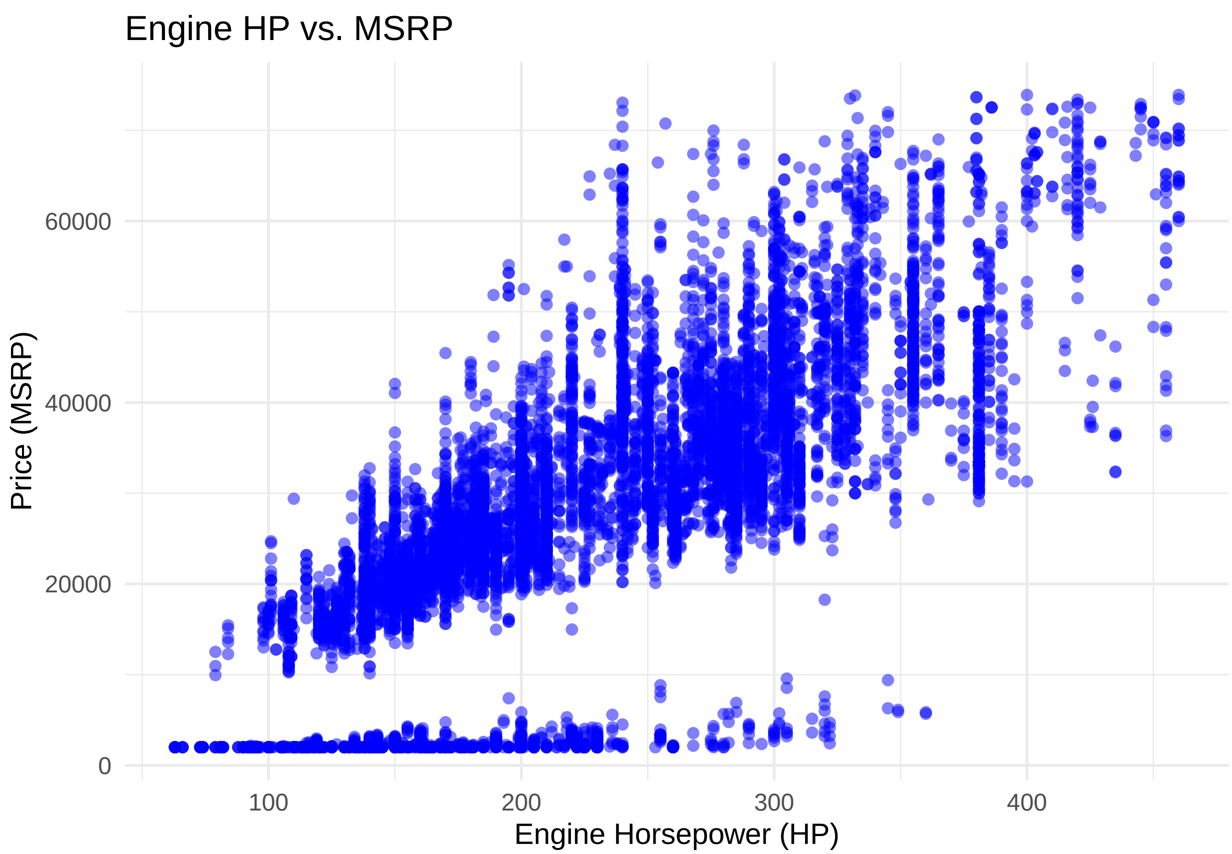
```
ggplot(cars_dataset, aes(x = Engine.HP)) +
  geom_histogram(binwidth = 20, fill = "lightgreen", color = "black") +
  labs(title = "Distribution of Engine Horsepower", x = "Engine HP", y = "Count") +
  theme_minimal()
```



4. Scatter Plot of Engine.HP vs. Price (MSRP)

Examines the relationship between horsepower and price, allowing analysis of how performance might influence vehicle pricing.

```
ggplot(cars_dataset, aes(x = Engine.HP, y = MSRP)) +
  geom_point(color = "blue", alpha = 0.5) +
  labs(title = "Engine HP vs. MSRP", x = "Engine Horsepower (HP)", y = "Price (MSRP)") +
  theme_minimal()
```



Commentary:

Through charts like histograms, scatter plots, and boxplots, we analyze variable distributions, relationships, and trends. This stage uncovers essential patterns, such as price and horsepower variation, offering an initial understanding of key car attributes that shape the dataset.

Stage 4: Analyzing the Price Variables

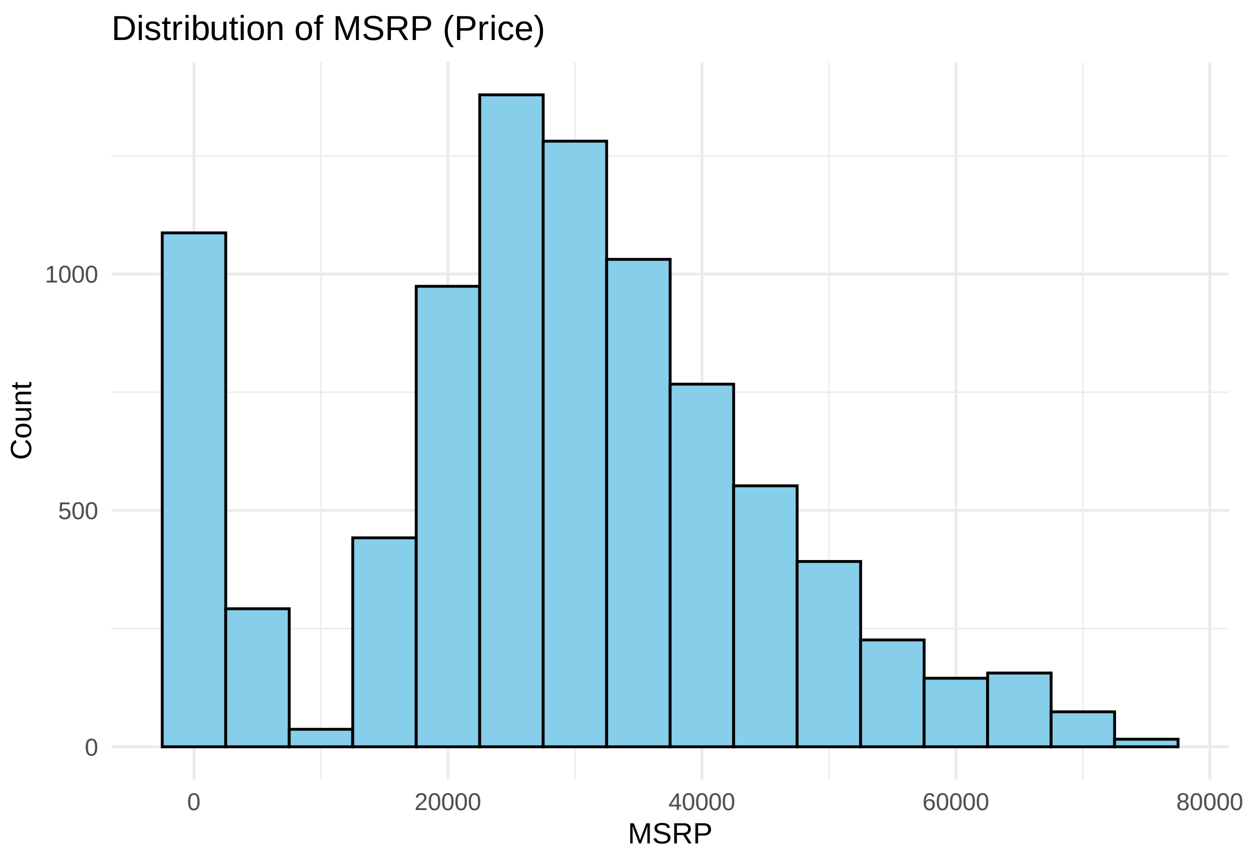
We analyze MSRP to understand price distribution, categorize cars by price range, compare prices across types, and explore price-related correlations.

4.1 Summary of Price Variable

1.Histogram of MSRP

The histogram shows car price distribution, highlighting concentration, skewness, and any outliers.

```
ggplot(cars_dataset, aes(x = MSRP)) +
  geom_histogram(binwidth = 5000, fill = "skyblue", color = "black") +
  labs(title = "Distribution of MSRP (Price)", x = "MSRP", y = "Count") +
  theme_minimal()
```



2.Summary Statistics for MSRP

Calculating mean, median, and variance of MSRP provides insight into average prices and price variability in the dataset.

```
mean_msrp <- mean(cars_dataset$MSRP, na.rm = TRUE)
median_msrp <- median(cars_dataset$MSRP, na.rm = TRUE)
var_msrp <- var(cars_dataset$MSRP, na.rm = TRUE)

print(paste("Mean MSRP:", mean_msrp))
```

```
## [1] "Mean MSRP: 28433.1955711219"
```

```
print(paste("Median MSRP:", median_msrp))
```

```
## [1] "Median MSRP: 28395"
```

```
print(paste("Variance of MSRP:", var_msrp))
```

```
## [1] "Variance of MSRP: 255028625.869883"
```

4.2 Grouping Cars by Price Range

1.Defining Price Ranges

Cars are divided into Low, Medium, High, and Luxury groups based on MSRP, allowing for feature comparisons across price segments.

```
cars_dataset$Price.Range <- cut(cars_dataset$MSRP,
                                breaks = c(0, 20000, 40000, 60000, Inf),
                                labels = c("Low", "Medium", "High", "Luxury"))
```

2.Summarizing by Price Range

Summary stats for each price range show differences in pricing within four price tiers.

```
summary_by_price_range <- aggregate(MSRP ~ Price.Range, data = cars_dataset, summary)
print(summary_by_price_range)
```

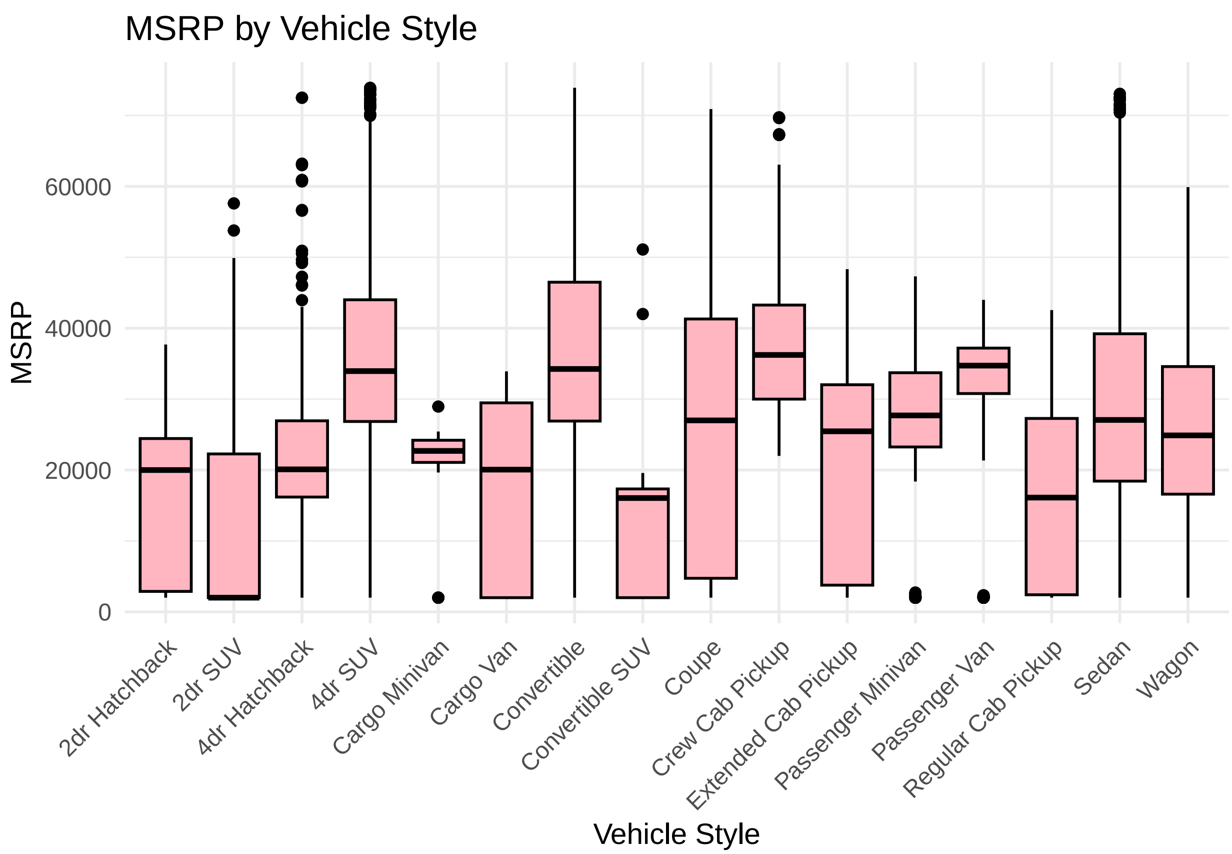
```
##   Price.Range MSRP.Min. MSRP.1st Qu. MSRP.Median MSRP.Mean MSRP.3rd Qu.
## 1         Low  2000.000     2000.000    2668.000  8023.807    16158.750
## 2      Medium 20015.000    24841.250   29097.000 29337.669    33758.750
## 3        High 40020.000    42765.000   46180.000 47264.335    50952.500
## 4      Luxury 60070.000    62685.000   64827.500 65494.525    68298.750
##   MSRP.Max.
## 1 20000.000
## 2 40000.000
## 3 60000.000
## 4 73905.000
```

4.3 Exploring Prices by Car Type

Boxplot for MSRP by Vehicle Style

Compares price ranges across vehicle styles, highlighting how different styles align with market value.

```
ggplot(cars_dataset, aes(x = Vehicle.Style, y = MSRP)) +
  geom_boxplot(fill = "lightpink", color = "black") +
  labs(title = "MSRP by Vehicle Style", x = "Vehicle Style", y = "MSRP") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



4.4 Correlation Analysis with Price

1.Correlation Matrix

We calculate a correlation matrix for numerical variables to find the top three factors most strongly associated with MSRP.

Select numerical columns

```
numeric_data <- cars_dataset[, sapply(cars_dataset, is.numeric)]
```

Correlation with MSRP

```
cor_matrix <- cor(numeric_data, use = "complete.obs")
cor_with_price <- cor_matrix["MSRP", ]
```

Top 3 correlated variables with MSRP (excluding MSRP itself)

```r
top_3_correlated <- sort(cor_with_price, decreasing = TRUE)[2:4]
print(top_3_correlated)
```

```
##        Engine.HP             Year Engine.Cylinders
##        0.7466106        0.7049611        0.2967746
```

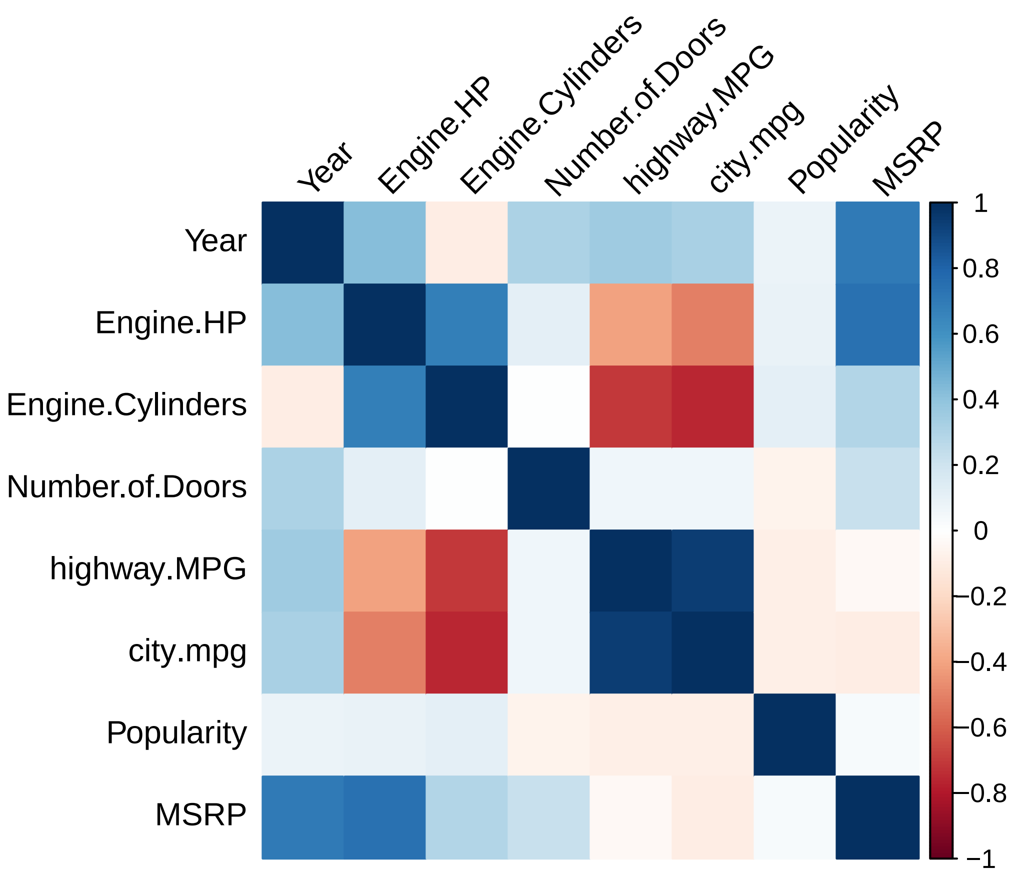
2.Correlation Plot

The correlation plot visually shows relationships among numerical variables, helping identify key factors positively or negatively impacting MSRP.

```r
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```r
corrplot(cor_matrix, method = "color", tl.col = "black", tl.srt = 45)
```



Commentary:

In this step, we analyze the price variable, MSRP, by examining its distribution, grouping cars by price ranges, and exploring pricing trends across vehicle types. Correlation analysis identifies influential factors, offering insights into what drives car prices.

Stage 5: Effect of Brand on Popularity and Price

This section examines how brand influences car price (MSRP) and popularity, helping to understand whether certain brands are perceived as luxury or are particularly popular among consumers.

1.Average MSRP by Brand

Calculates each brand's average price, distinguishing luxury from budget-friendly options.

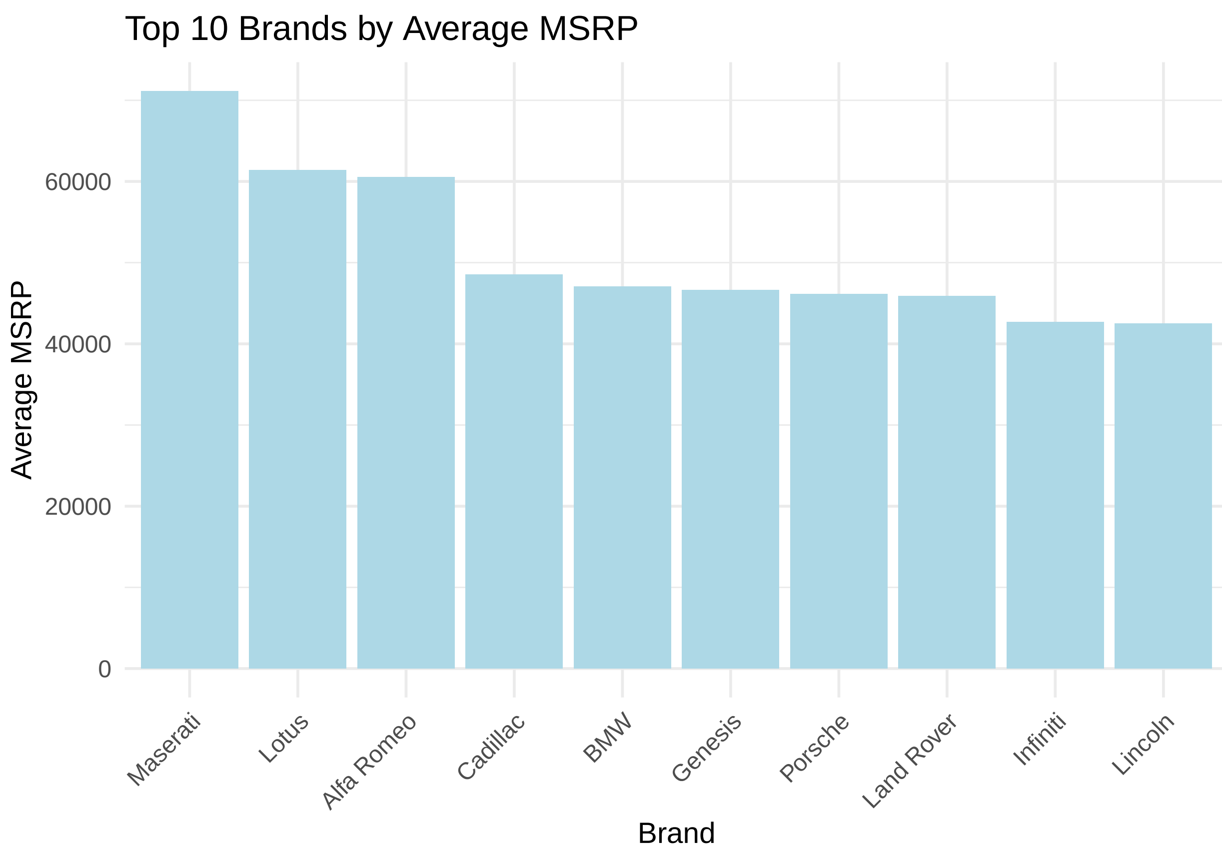
The average price per brand to see which brands are positioned as luxury versus affordable.

```
avg_price_by_brand <- aggregate(MSRP ~ Make, data = cars_dataset, mean)
avg_price_by_brand <- avg_price_by_brand[order(-avg_price_by_brand$MSRP), ]
```

Top 10 Brands by Average MSRP (Bar Plot)

Shows top 10 brands by average price, highlighting luxury brands.

```
top_10_avg_price <- head(avg_price_by_brand, 10)
ggplot(data = top_10_avg_price, aes(x = reorder(Make, -MSRP), y = MSRP)) +
  geom_bar(stat = "identity", fill = "lightblue") +
  labs(title = "Top 10 Brands by Average MSRP", x = "Brand", y = "Average MSRP") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



2.Average Popularity by Brand

Measures brand popularity, identifying those with strong market appeal.

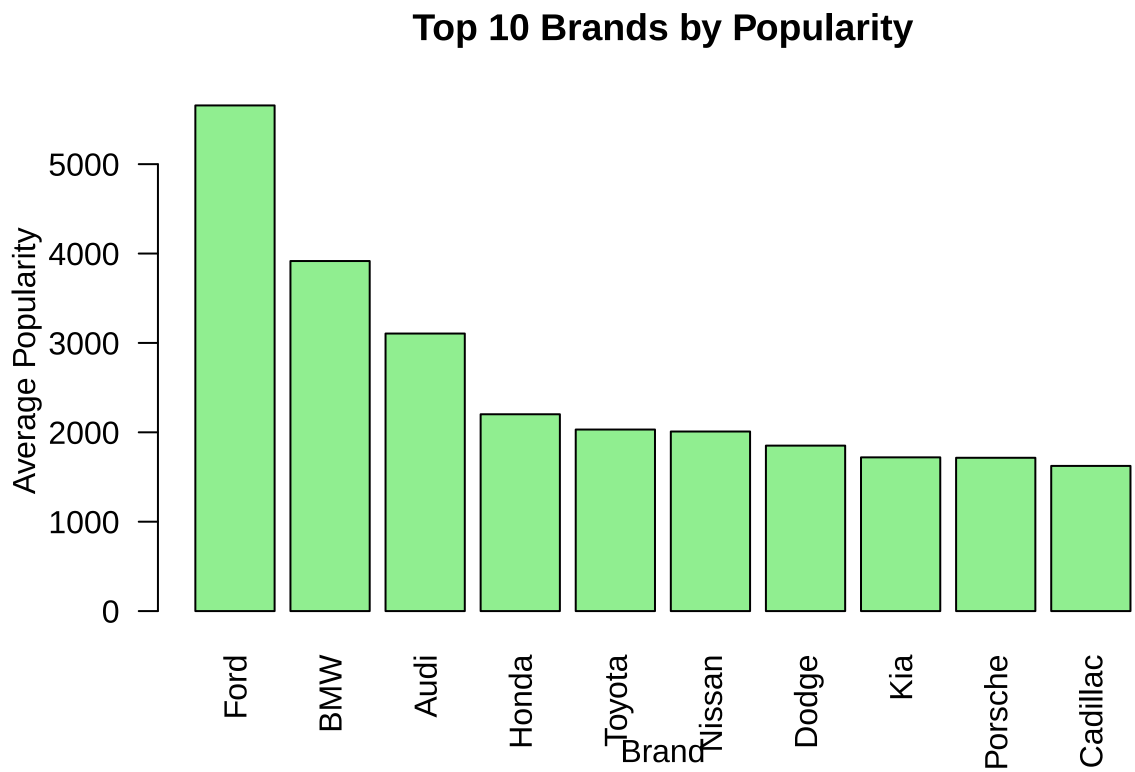
The average popularity per brand to see which brands are the most popular.

```
avg_popularity_by_brand <- aggregate(Popularity ~ Make, data = cars_dataset, mean)
avg_popularity_by_brand <- avg_popularity_by_brand[order(-avg_popularity_by_brand$Popularity), ]
```

Top 10 Brands by Average Popularity (Bar Plot)

Displays the most popular brands based on average consumer favorability.

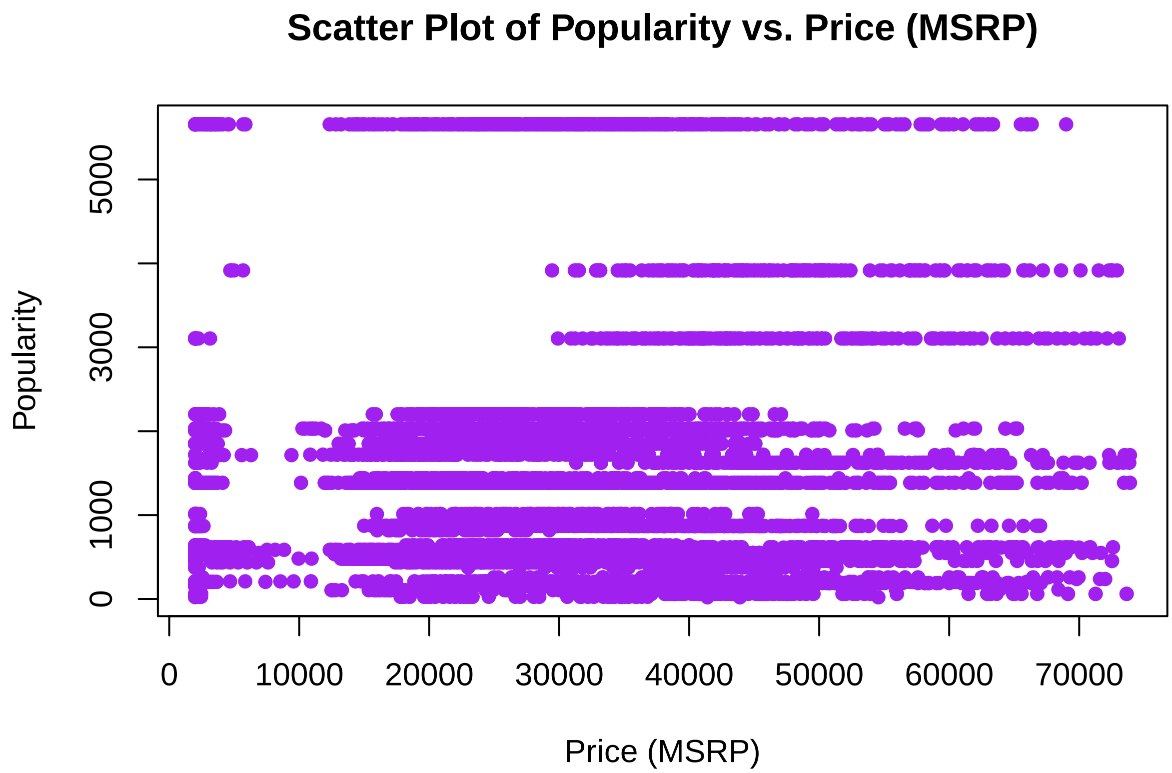
```
top_10_popularity <- head(avg_popularity_by_brand, 10)
barplot(top_10_popularity$Popularity,
        names.arg = top_10_popularity$Make,
        main = "Top 10 Brands by Popularity",
        xlab = "Brand",
        ylab = "Average Popularity",
        col = "lightgreen",
        las = 2)
```



3.Scatter Plot: Popularity vs. Price (MSRP)

Examines if higher-priced cars are more or less popular, identifying trends in consumer preferences across price ranges.

```r
plot(cars_dataset$MSRP, cars_dataset$Popularity,
     main = "Scatter Plot of Popularity vs. Price (MSRP)",
     xlab = "Price (MSRP)",
     ylab = "Popularity",
     col = "purple",
     pch = 16)
```



Commentary:

Analyzing the brand's impact on both popularity and price highlights market positioning and consumer preferences. By calculating average prices and popularity by brand, we differentiate luxury brands from budget options. The relationship between price and popularity is also explored in this step.